Flores González Luis Brandon, 312218342, iluis@ciencias.unam.mx
Garcia Argueta Jaime Daniel, 312104739, jaimegarciaargueta@ciencias.unam.mx

- Primero **se crea un tipo de dato binario**. Este contendrá BaseUno, Cero Binario y Uno Binario
- Se crea una **instancia de la clase show** para mostrar el tipo de dato binario como cadena de caracteres. Esto se hace de manera recursiva, tomando la definición del bit menos significativo que se vio en clase.
- Se crea una **función sucesor** que recibe un binario y devuelve su sucesor. Solo prende el bit menos significativo si está apagado y apaga los bits que estén prendidos antes del primero que sea uno(comenzando del menos significativo).
- Se crea una **función suma** la cual suma dos binarios. Esta se hace recursivamente, solo suma bit a bit comenzando desde el menos significativo.
- Se crea una **función predecesor.** Esta solo apaga el primer bit que sea uno y prende los anteriores(desde el bit menos significativo). Se hace recursivamente.
- Se crea una **función natBilLista** la cual recibe una natural y devuelve una lista de enteros representando su notación binaria. Se usa una **función auxiliar makeBinList** que se describira más abajo.
- Se crea una **función sumaBinLista** la cual recibe dos binarios representados en lista y devuelve su suma en el tipo de datos Binario. Se usan las **funciones natABin, binANat y binListToBin** para reutilizar código.

Se hicieron los puntos extra que son las funciones natABin y binANat que son las más importantes.

- Se crea la **función natAbin** que recibe un natural y devuelve un binario. Esta crea un binario a partir de los tipos de datos de binario. Se hace recursivamente y se usa if else. Además se usa even para saber si es par un número y div para dividir.
- Se crea la **función binANat** que recibe un tipo de dato Binario y devuelve su representación en un número entero. Esta usa una **función auxiliar binANatPow**.

Aparte de los puntos extra, se crearon **funciones auxiliares** para facilitar el trabajo a las funciones principales descritas en la práctica.

- Se crea la **función makeBinList** la cual recibe un binario y devuelve una lista de 0's y 1's de sus bits respectivos.
- Se crea la **función binListaABin** la cual recibe una lista de un binario y devuelve el mismo número pero como tipo Binario e invertido.
- Se crea la **función binListToBin** la cual recibe un binario en su representación de lista y llama a la función binListaABin con la lista invertida para convertir el número a un tipo Binario.
- Se crea la **función invertList** la cual recibe una lista de enteros y devuelve la misma lista pero invertida.
- Se crea la **función binANatPow** la cual Recibe un binario, un exponente y devuelve el número 2 elevado al exponente si el bit menos significativo del binario es uno y devuelve cero si el bit menos significativo es cero.

Finalmente **se pueden poner los mismos ejemplos que se especifican en la práctica** para probar las funciones, ya que se hizo de la manera especificada sólo hubo un cambio en el nombre de **binANatPot**, en esta práctica la llamamos **binANatPow**. Además mostrará los resultados esperados.